Lecture 5: Conditions

Comp 102 (sec: A,B)

Fakhir Shaheen

Forman Christian University

Recap

Pop Quiz: True or False?

Let `x = 10` and `y = 3`

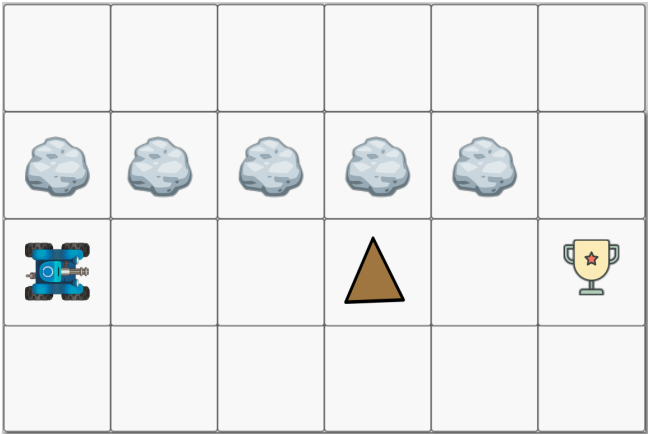
1. `x > y`
2. `x == (y * 3)`
3. `not (x < 5)`
4. `x >= 10 and y < 5`

Pop Quiz: True or False?

Let `x = 10` and `y = 3`

1. `x > y` — **True**
2. `x == (y * 3)` — **False** (10 vs 9)
3. `not (x < 5)` — **True** (not False)
4. `x >= 10 and y < 5` — **True** (True and True)

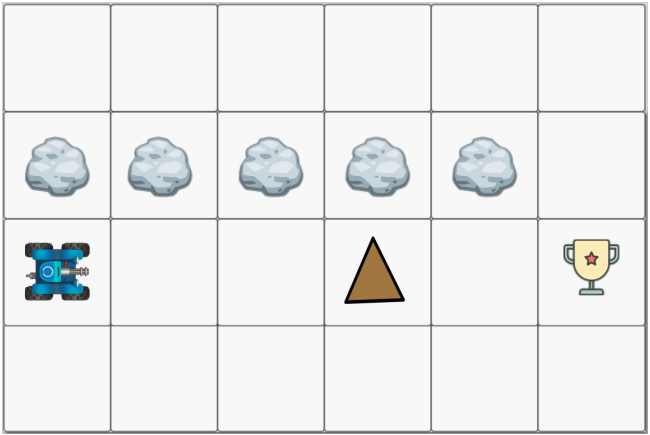
- **Commands Available:**
  - `forward()` $\times 1$
  - `remove()` $\times 1$
  - `repeat n:` $\times 1$
  - `if dirt:` $\times 1$



- **Commands Available:**
  - `forward()` $\times 1$
  - `remove()` $\times 1$
  - `repeat n:` $\times 1$
  - `if dirt:` $\times 1$

- **Solution:**

```
1 repeat 5:
2     if dirt:
3         remove()
4     forward()
```



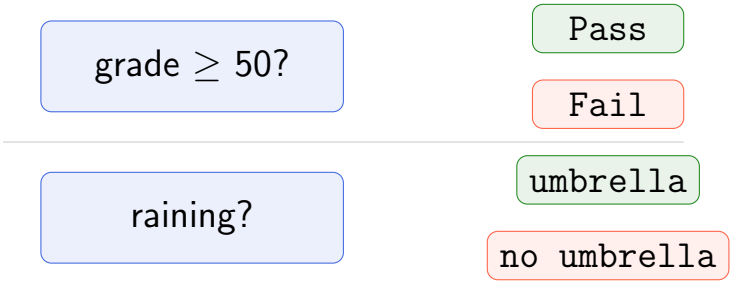
Why Conditions?

Programs need to make **decisions**!



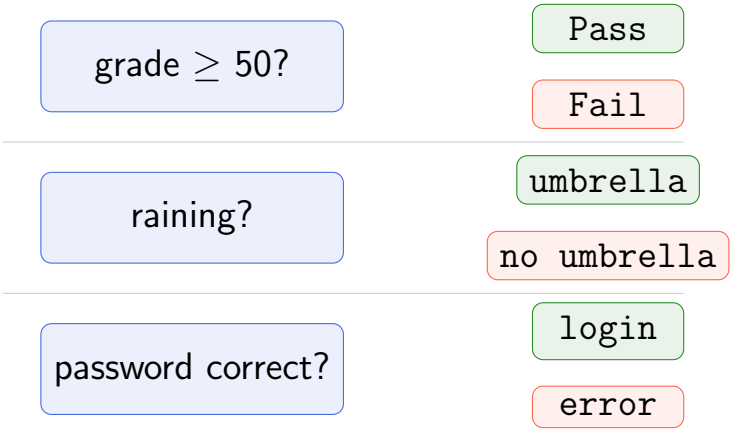
Why Conditions?

Programs need to make **decisions**!



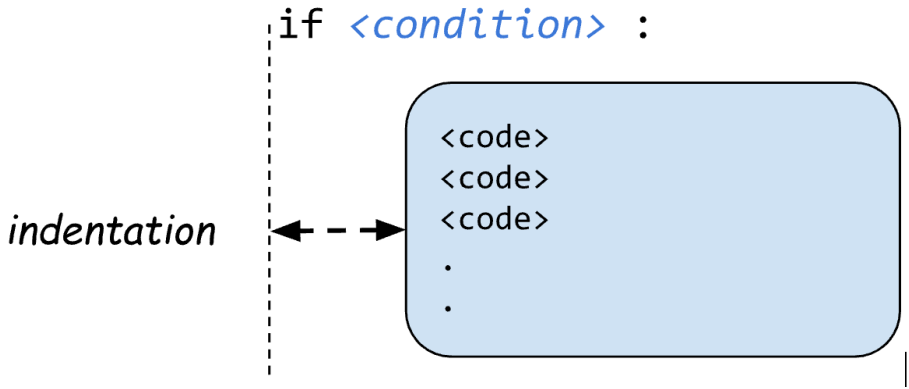
Why Conditions?

Programs need to make **decisions**!

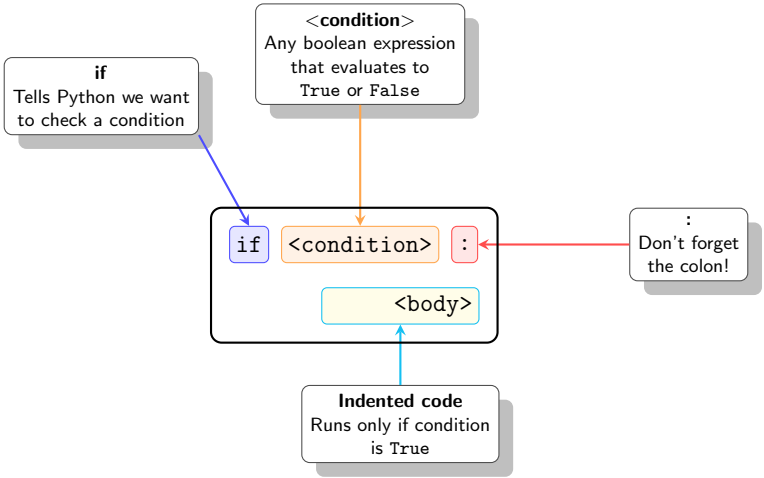


The `if` Statement

The if command:

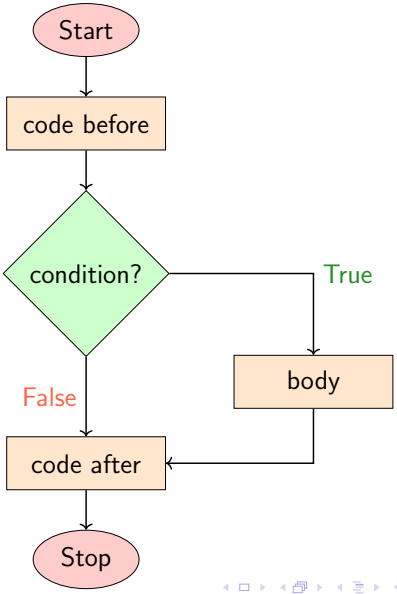


`if` Syntax


if
Tells Python we want to check a condition
<condition>
Any boolean expression that evaluates to True or False
if
<condition>
:
:
Don't forget the colon!
<body>
Indented code
Runs only if condition is True

`if` Flowchart

- Evaluate the condition
  - If **True** → execute the body
  - If **False** → skip the body
- Continue with code after



Example: Even Number

```
1 num = int(input("Enter a number: "))
2 if num % 2 == 0:
3     print(f"{num} is Even")
4 print("Bye")
```

Example: Even Number

```
1 num = int(input("Enter a number: "))
2 if num % 2 == 0:
3     print(f"{num} is Even")
4 print("Bye")
```

Trace 1: `num = 6`

6 % 2 == 0 → `True` → prints `"6 is Even"`

Then prints `"Bye"`

Example: Even Number

```
1 num = int(input("Enter a number: "))
2 if num % 2 == 0:
3     print(f"{num} is Even")
4 print("Bye")
```

Trace 1: `num = 6`

6 % 2 == 0 → `True` → prints `"6 is Even"`

Then prints `"Bye"`

Trace 2: `num = 7`

7 % 2 == 0 → `False` → *skip body*

Then prints `"Bye"`

Example: Even Number

```
1 num = int(input("Enter a number: "))
2 if num % 2 == 0:
3     print(f"{num} is Even")
4 print("Bye")
```

Trace 1: `num = 6`

6 % 2 == 0 → `True` → prints `"6 is Even"`

Then prints `"Bye"`

Trace 2: `num = 7`

7 % 2 == 0 → `False` → *skip body*

Then prints `"Bye"`

Notice: `"Bye"` prints in **BOTH** cases, it's **outside** the `if`!

Example: Temperature

```
1 temp = int(input("Temperature: "))
2 if temp > 30:
3     print("It's hot today!")
4     print("Stay hydrated!")
5 print("Have a good day!")
```

Example: Temperature

```
1 temp = int(input("Temperature: "))
2 if temp > 30:
3     print("It's hot today!")
4     print("Stay hydrated!")
5 print("Have a good day!")
```

Trace 1: `temp = 35`

35 > 30 → `True` → prints **both** body lines

Then prints `"Have a good day!"`

Example: Temperature

```
1 temp = int(input("Temperature: "))
2 if temp > 30:
3     print("It's hot today!")
4     print("Stay hydrated!")
5 print("Have a good day!")
```

Trace 1: `temp = 35`

35 > 30 → `True` → prints **both** body lines

Then prints `"Have a good day!"`

Trace 2: `temp = 20`

20 > 30 → `False` → *skip both body lines*

Then prints `"Have a good day!"`

You Try

What is the output?

```
1 age = int(input("Your age: "))
2 if age >= 18:
3     print("You can vote!")
4 print("Thanks for checking.")
```

You Try

What is the output?

```
1 age = int(input("Your age: "))
2 if age >= 18:
3     print("You can vote!")
4 print("Thanks for checking.")
```

If `age = 15`:

`15 >= 18` → `False` → only prints `"Thanks for checking."`

You Try

What is the output?

```
1 age = int(input("Your age: "))
2 if age >= 18:
3     print("You can vote!")
4 print("Thanks for checking.")
```

If `age = 15`:

`15 >= 18` → `False` → only prints `"Thanks for checking."`

If `age = 18`:

`18 >= 18` → `True` → prints `"You can vote!"`

Then prints `"Thanks for checking."`

Tricky!

Indentation changes the meaning!

Code A

```
1 x = 5
2 if x > 10:
3     print("Big")
4     print("Number")
```

Code B

```
1 x = 5
2 if x > 10:
3     print("Big")
4 print("Number")
```

Tricky!

Indentation changes the meaning!

Code A

```
1 x = 5
2 if x > 10:
3     print("Big")
4     print("Number")
```

Output: *(nothing)*

Both prints are **inside** the if

Code B

```
1 x = 5
2 if x > 10:
3     print("Big")
4 print("Number")
```

Output: `Number`

Second print is **outside** the if

Tricky!

Indentation changes the meaning!

Code A

```
1 x = 5
2 if x > 10:
3     print("Big")
4     print("Number")
```

Output: *(nothing)*

Both prints are **inside** the if

Code B

```
1 x = 5
2 if x > 10:
3     print("Big")
4 print("Number")
```

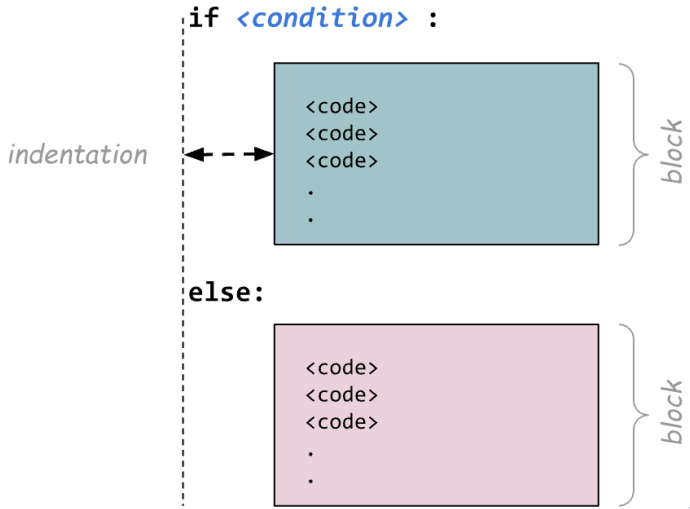
Output: `Number`

Second print is **outside** the if

Same code, different indentation, different behavior!

The `if/else` Statement

The **if/else** command:



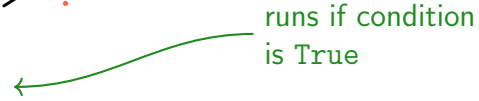
if/else Syntax

```
if  <condition> :
        <body1>
else:
        <body2>
```

if/else Syntax

```
if   <condition> :
        <body1>

else:
        <body2>
```

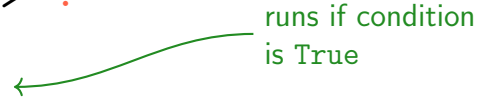




if/else Syntax

```
if   <condition> :
        <body1>

else:
        <body2>
```

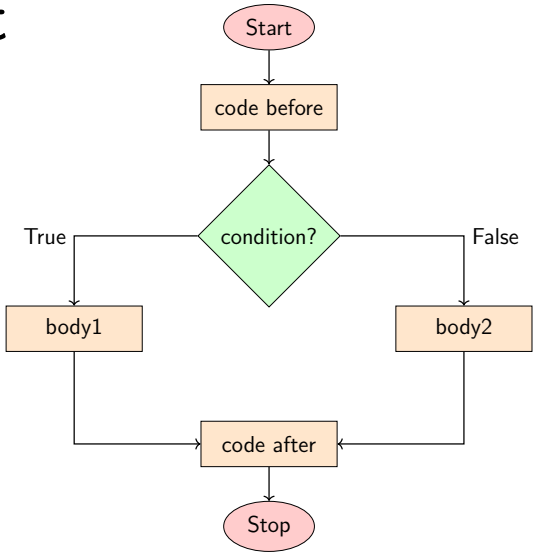




Rule: Exactly **ONE** body always runs. **Never both, never neither.**

`if/else` Flowchart

- Evaluate the condition
- If **True** → execute body1
- If **False** → execute body2
- Continue with code after



Example: Weather

```
1 answer = input("Is it raining? (yes/no): ")
2 if answer == "yes":
3     print("Take an umbrella!")
4 else:
5     print("Enjoy the sunshine!")
6 print("Have a great day!")
```

Example: Weather

```
1 answer = input("Is it raining? (yes/no): ")
2 if answer == "yes":
3     print("Take an umbrella!")
4 else:
5     print("Enjoy the sunshine!")
6 print("Have a great day!")
```

Trace 1: `answer = "yes"`

`"yes" == "yes"` → `True` → `"Take an umbrella!"`

Then: `"Have a great day!"`

Trace 2: `answer = "no"`

`"no" == "yes"` → `False` → `"Enjoy the sunshine!"`

Example: Pass/Fail

```
1 score = int(input("Enter your score: "))
2 if score >= 50:
3     print("You passed!")
4 else:
5     print("You failed. Try again!")
```

Example: Pass/Fail

```
1 score = int(input("Enter your score: "))
2 if score >= 50:
3     print("You passed!")
4 else:
5     print("You failed. Try again!")
```

Trace 1: `score = 72`

72 >= 50 → `True` → `"You passed!"`

Example: Pass/Fail

```
1 score = int(input("Enter your score: "))
2 if score >= 50:
3     print("You passed!")
4 else:
5     print("You failed. Try again!")
```

Trace 1: `score = 72`

72 >= 50 → `True` → `"You passed!"`

Trace 2: `score = 35`

35 >= 50 → `False` → `"You failed. Try again!"`

You Try

Write a program that tells whether a number is **even** or **odd**.

You Try

Write a program that tells whether a number is **even** or **odd**.

```
num = int(input("Enter a number: "))
if num % 2 == 0:
    print(f"{num} is even")
else:
    print(f"{num} is odd")
```

You Try: Fix the Bug ✗

```
x = int(input("Enter x: "))
y = int(input("Enter y: "))
if x == y:
    print(x, "and", y, "are equal!")
print("They match!")
```

You Try: Fix the Bug ✗

```
1 x = int(input("Enter x: "))
2 y = int(input("Enter y: "))
3 if x == y:
4     print(x, "and", y, "are equal!")
5 print("They match!")
```

Problem: `"They match!"` prints even when `x != y`

It's **outside** the `if` block.

You Try: Fix the Bug ✗

```
x = int(input("Enter x: "))
y = int(input("Enter y: "))
if x == y:
    print(x, "and", y, "are equal!")
print("They match!")
```

Problem: `"They match!"` prints even when `x != y`

It's **outside** the `if` block.

Fix: indent it into the `if` body, or restructure with `if/else`:

```
if x == y:
    print(x, "and", y, "are equal!")
    print("They match!")
else:
    print("They are different!")
```

The `elif` Chain

Multiple ifs — A Problem

```
1 score = 85
2 if score >= 90:
3     grade = "A"
4 if score >= 80:
5     grade = "B"
6 if score >= 70:
7     grade = "C"
8 if score >= 60:
9     grade = "D"
```

Multiple `if`s — A Problem

```
1 score = 85
2 if score >= 90:
3     grade = "A"
4 if score >= 80:
5     grade = "B"
6 if score >= 70:
7     grade = "C"
8 if score >= 60:
9     grade = "D"
```

Trace: `score = 85`

`85 >= 90` → `False` (skip)

Multiple `if`s — A Problem

```
1 score = 85
2 if score >= 90:
3     grade = "A"
4 if score >= 80:
5     grade = "B"
6 if score >= 70:
7     grade = "C"
8 if score >= 60:
9     grade = "D"
```

Trace: `score = 85`

`85 >= 90` → `False` (skip)

`85 >= 80` → `True` → grade = `"B"`

`85 >= 70` → `True` → grade = `"C"`

`85 >= 60` → `True` → grade = `"D"`

Multiple `if`s — A Problem

```
1 score = 85
2 if score >= 90:
3     grade = "A"
4 if score >= 80:
5     grade = "B"
6 if score >= 70:
7     grade = "C"
8 if score >= 60:
9     grade = "D"
```

Trace: `score = 85`

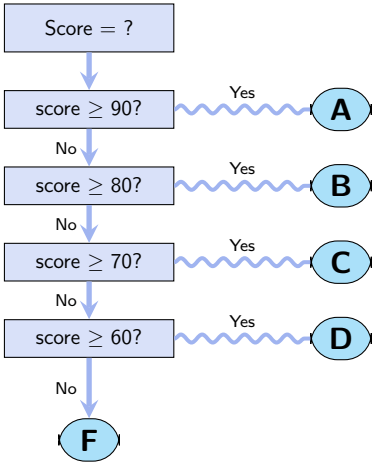
`85 >= 90` → `False` (skip)
`85 >= 80` → `True` → grade = `"B"`
`85 >= 70` → `True` → grade = `"C"`
`85 >= 60` → `True` → grade = `"D"`

Wrong! grade = `"D"` for a score of 85! All conditions are checked **independently**.

`elif` – The Waterfall Model

Think of `elif` chains like a **waterfall**:

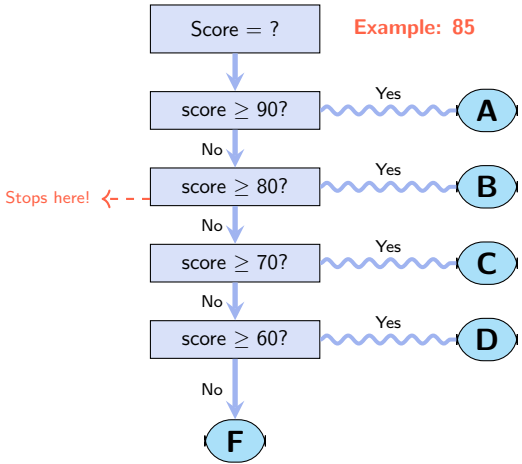
1. Water (your program) starts at the top
2. At each level, it checks: "Can I stop here?"
3. **First True** = water stops at that pool
4. If none True, water reaches the bottom (`else`)



`elif` – The Waterfall Model

Think of `elif` chains like a **waterfall**:

1. Water (your program) starts at the top
2. At each level, it checks: "Can I stop here?"
3. **First True** = water stops at that pool
4. If none True, water reaches the bottom (`else`)



First True Wins!

Once Python finds a `True` condition:

1. Executes that block of code
2. **Skips** all remaining `elif` and `else` blocks
3. Continues with code **after** the entire structure

```
1  score = 85
2  if score >= 90:    # False - skip
3      print("A")
4  elif score >= 80: # True - Run this!
5      print("B")
6  elif score >= 70: # Never checked
7      print("C")
8  else:              # Never checked
9      print("F")
10 print("Done grading")  # Always runs
```

Example: Letter Grade

```
1  score = int(input("Enter score: "))
2  if score >= 90:
3      grade = "A"
4  elif score >= 80:
5      grade = "B"
6  elif score >= 70:
7      grade = "C"
8  elif score >= 60:
9      grade = "D"
10 else:
11     grade = "F"
12 print(f"Your grade is {grade}")
```

Example: Letter Grade

```
1  score = int(input("Enter score: "))
2  if score >= 90:
3      grade = "A"
4  elif score >= 80:
5      grade = "B"
6  elif score >= 70:
7      grade = "C"
8  elif score >= 60:
9      grade = "D"
10 else:
11     grade = "F"
12 print(f"Your grade is {grade}")
```

Trace 1: `score = 85`

Example: Letter Grade

```
1  score = int(input("Enter score: "))
2  if score >= 90:
3      grade = "A"
4  elif score >= 80:
5      grade = "B"
6  elif score >= 70:
7      grade = "C"
8  elif score >= 60:
9      grade = "D"
10 else:
11     grade = "F"
12 print(f"Your grade is {grade}")
```

Trace 1: `score = 85`

85 >= 90 → False

85 >= 80 → True → grade = "B" → **STOP**

Output: "Your grade is B"

Example: Letter Grade

```
1  score = int(input("Enter score: "))
2  if score >= 90:
3      grade = "A"
4  elif score >= 80:
5      grade = "B"
6  elif score >= 70:
7      grade = "C"
8  elif score >= 60:
9      grade = "D"
10 else:
11     grade = "F"
12 print(f"Your grade is {grade}")
```

Trace 1: `score = 85`

85 >= 90 → False

85 >= 80 → True → grade = "B" → **STOP**

Output: "Your grade is B"

Trace 2: `score = 55`

All conditions False → else → grade = "F"

Output: "Your grade is F"

Example: Day Greeting

```
1  day = input("What day is it? ")
2  if day == "Monday":
3      print("Start of the week!")
4  elif day == "Friday":
5      print("Almost weekend!")
6  elif day == "Saturday" or \
7       day == "Sunday":
8      print("Weekend! Relax!")
9  else:
10     print("Keep going!")
```

Example: Day Greeting

```
1  day = input("What day is it? ")
2  if day == "Monday":
3      print("Start of the week!")
4  elif day == "Friday":
5      print("Almost weekend!")
6  elif day == "Saturday" or \
7       day == "Sunday":
8      print("Weekend! Relax!")
9  else:
10     print("Keep going!")
```

Trace: `day = "Friday"`

`"Friday" == "Monday"` → `False`

`"Friday" == "Friday"` → `True` →

`"Almost weekend!"` → **STOP**

Example: Day Greeting

```
1  day = input("What day is it? ")
2  if day == "Monday":
3      print("Start of the week!")
4  elif day == "Friday":
5      print("Almost weekend!")
6  elif day == "Saturday" or \
7       day == "Sunday":
8      print("Weekend! Relax!")
9  else:
10     print("Keep going!")
```

Trace: `day = "Friday"`
`"Friday" == "Monday"` → `False`
`"Friday" == "Friday"` → `True` →
`"Almost weekend!"` → **STOP**
Trace: `day = "Wednesday"`

All conditions `False` → `else` →
`"Keep going!"`

You Try

Write an `elif` chain for temperature categories:

- Above 35: `"Extremely hot!"`
- 25–35: `"Nice weather"`
- 15–24: `"A bit cool"`
- Below 15: `"Cold! Wear a jacket"`

You Try

Write an `elif` chain for temperature categories:

- Above 35: `"Extremely hot!"`
- 25–35: `"Nice weather"`
- 15–24: `"A bit cool"`
- Below 15: `"Cold! Wear a jacket"`

```
1 temp = int(input("Temperature: "))
2 if temp > 35:
3     print("Extremely hot!")
4 elif temp >= 25:
5     print("Nice weather")
6 elif temp >= 15:
7     print("A bit cool")
8 else:
9     print("Cold! Wear a jacket")
```

Tricky!

What happens if we reverse the order?

```
1 score = 95
2 if score >= 60:
3     grade = "D"
4 elif score >= 70:
5     grade = "C"
6 elif score >= 80:
7     grade = "B"
8 elif score >= 90:
9     grade = "A"
```

Tricky!

What happens if we reverse the order?

```
1 score = 95
2 if score >= 60:
3     grade = "D"
4 elif score >= 70:
5     grade = "C"
6 elif score >= 80:
7     grade = "B"
8 elif score >= 90:
9     grade = "A"
```

Trace: `score = 95`
95 >= 60 → `True` → grade = "D" → **STOP**
A score of 95 gets a D!

Tricky!

What happens if we reverse the order?

```
1 score = 95
2 if score >= 60:
3     grade = "D"
4 elif score >= 70:
5     grade = "C"
6 elif score >= 80:
7     grade = "B"
8 elif score >= 90:
9     grade = "A"
```

Trace: `score = 95`
95 >= 60 → True → grade = "D" → **STOP**
A score of 95 gets a D!

Rule: Put the **MOST SPECIFIC** (strictest) condition **FIRST**.

Nested `if` Statements

What is Nesting?

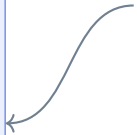
An `if` inside another `if`

```
if condition1:
    if condition2:
        body (both True)
```

What is Nesting?

An `if` inside another `if`

```
if condition1:
    if condition2:
        body (both True)
```



Outer must be `True` before inner is even checked

Example: Ride Check

```
1  height = int(input("Height in cm: "))
2  if height >= 120:
3      age = int(input("Age: "))
4      if age >= 8:
5          print("You can go on the ride!")
6      else:
7          print("Must be 8 or older.")
8  else:
9      print("Must be at least 120 cm.")
```

Example: Ride Check

```
1  height = int(input("Height in cm: "))
2  if height >= 120:
3      age = int(input("Age: "))
4      if age >= 8:
5          print("You can go on the ride!")
6      else:
7          print("Must be 8 or older.")
8  else:
9      print("Must be at least 120 cm.")
```

Trace 1: `height=130, age=10`
`130 >= 120` → `True`, `10 >= 8` → `True` → `"You can go on the ride!"`

Example: Ride Check

```
1  height = int(input("Height in cm: "))
2  if height >= 120:
3      age = int(input("Age: "))
4      if age >= 8:
5          print("You can go on the ride!")
6      else:
7          print("Must be 8 or older.")
8  else:
9      print("Must be at least 120 cm.")
```

Trace 1: `height=130, age=10`
130 >= 120 → True, 10 >= 8 → True → "You can go on the ride!"

Trace 2: `height=130, age=6`
130 >= 120 → True, 6 >= 8 → False → "Must be 8 or older."

Example: Ride Check

```
1 height = int(input("Height in cm: "))
2 if height >= 120:
3     age = int(input("Age: "))
4     if age >= 8:
5         print("You can go on the ride!")
6     else:
7         print("Must be 8 or older.")
8 else:
9     print("Must be at least 120 cm.")
```

Trace 1: `height=130, age=10`
130 >= 120 → True, 10 >= 8 → True → "You can go on the ride!"

Trace 2: `height=130, age=6`
130 >= 120 → True, 6 >= 8 → False → "Must be 8 or older."

Trace 3: `height=100`
100 >= 120 → False → "Must be at least 120 cm." (inner if never reached)

Example: Login System

```
1  username = input("Username: ")
2  if username == "admin":
3      password = input("Password: ")
4      if password == "secret123":
5          print("Welcome, admin!")
6      else:
7          print("Wrong password!")
8  else:
9      print("Unknown user.")
```

Example: Login System

```
1  username = input("Username: ")
2  if username == "admin":
3      password = input("Password: ")
4      if password == "secret123":
5          print("Welcome, admin!")
6      else:
7          print("Wrong password!")
8  else:
9      print("Unknown user.")
```

Trace 1: `user="admin"`, `pass="secret123"`
Both `True` → `"Welcome, admin!"`

Example: Login System

```
1  username = input("Username: ")
2  if username == "admin":
3      password = input("Password: ")
4      if password == "secret123":
5          print("Welcome, admin!")
6      else:
7          print("Wrong password!")
8  else:
9      print("Unknown user.")
```

Trace 1: `user="admin"`, `pass="secret123"`
Both `True` → `"Welcome, admin!"`

Trace 2: `user="admin"`, `pass="wrong"`
Outer `True`, inner `False` → `"Wrong password!"`

Trace 3: `user="bob"`
Outer `False` → `"Unknown user."`
(never asked for password)

You Try

What is the output?

```
x = 15
y = 10
if x > y:
    if x > 20:
        print("A")
    else:
        print("B")
else:
    print("C")
print("D")
```

You Try

What is the output?

```
1  x = 15
2  y = 10
3  if x > y:
4      if x > 20:
5          print("A")
6      else:
7          print("B")
8  else:
9      print("C")
10 print("D")
```

Trace:

15 > 10 → True

15 > 20 → False → else

You Try

What is the output?

```
1  x = 15
2  y = 10
3  if x > y:
4      if x > 20:
5          print("A")
6      else:
7          print("B")
8  else:
9      print("C")
10 print("D")
```

Trace:

15 > 10 → True

15 > 20 → False → else

Output:

B

D

Nesting vs. and

Nested

```
1 if height >= 120:
2     if age >= 8:
3         print("OK!")
4     else:
5         print("Too young")
6 else:
7     print("Too short")
```

Nesting vs. and

Nested

```
1 if height >= 120:
2     if age >= 8:
3         print("OK!")
4     else:
5         print("Too young")
6 else:
7     print("Too short")
```

Flat (with and)

```
1 if height >=
  120 and age >= 8:
2     print("OK!")
3 else:
4     print("Not allowed")
```

Nesting vs. and

Nested

```
1 if height >= 120:
2     if age >= 8:
3         print("OK!")
4     else:
5         print("Too young")
6 else:
7     print("Too short")
```

Flat (with and)

```
1 if height >=
  120 and age >= 8:
2     print("OK!")
3 else:
4     print("Not allowed")
```

Tip: Nesting is better when you want **different messages** for each failure.

Combining Conditions

Using and in if

```
1  age = int(input("Age: "))
2  gpa = float(input("GPA: "))
3  if age >= 18 and gpa >= 3.0:
4      print("OK for scolrship")
5  else:
6      print("Not eligible.")
```

Using and in if

```
1 age = int(input("Age: "))
2 gpa = float(input("GPA: "))
3 if age >= 18 and gpa >= 3.0:
4     print("OK for scolrship")
5 else:
6     print("Not eligible.")
```

Trace 1: age=20, gpa=3.5

20 >= 18 → True and 3.5 >= 3.0 → True

True and True → True → "Eligible!"

Using and in if

```
1 age = int(input("Age: "))
2 gpa = float(input("GPA: "))
3 if age >= 18 and gpa >= 3.0:
4     print("OK for scolrship")
5 else:
6     print("Not eligible.")
```

Trace 1: age=20, gpa=3.5

20 >= 18 → True and 3.5 >= 3.0 → True

True and True → True → "Eligible!"

Trace 2: age=20, gpa=2.5

True and False → False → "Not eligible."

and: **BOTH** must be True

Using `or` in `if`

```
1  day = input("What day? ")
2  if day == "Saturday" or \
3     day == "Sunday":
4      print("It's the weekend!")
5  else:
6      print("It's a weekday.")
```

Using `or` in `if`

```
1  day = input("What day? ")
2  if day == "Saturday" or \
3     day == "Sunday":
4      print("It's the weekend!")
5  else:
6      print("It's a weekday.")
```

Trace 1: `day = "Saturday"`
`True or False` → `True` → `"It's the weekend!"`

Using `or` in `if`

```
1  day = input("What day? ")
2  if day == "Saturday" or \
3     day == "Sunday":
4      print("It's the weekend!")
5  else:
6      print("It's a weekday.")
```

Trace 1: `day = "Saturday"`
`True` or `False` → `True` → `"It's the weekend!"`

Trace 2: `day = "Monday"`
`False` or `False` → `False` → `"It's a weekday."`

or: **at least ONE** must be True

Using `not` in `if`

```
1 logged_in = False
2 if not logged_in:
3     print("Please log in first!")
```

`not False` → `True` → prints the message

Using `not` in `if`

```
1 logged_in = False
2 if not logged_in:
3     print("Please log in first!")
```

`not False` → `True` → prints the message

Also useful with strings:

```
1 name = input("Enter name: ")
2 if not name:
3     print("You didn't enter a name!"
```

Empty string `""` is falsy, so `not ""` → `True`

Example: Movie Ticket

```
1  age = int(input("Age: "))
2  is_student = input("Student? ")
3  if age < 12 or age >= 65 \
4     or is_student == "yes":
5      print("Discounted ticket: $8")
6  else:
7      print("Regular ticket: $15")
```

Example: Movie Ticket

```
1  age = int(input("Age: "))
2  is_student = input("Student? ")
3  if age < 12 or age >= 65 \
4     or is_student == "yes":
5      print("Discounted ticket: $8")
6  else:
7      print("Regular ticket: $15")
```

Trace 1: `age=10, student="no"`

`True` or `False` or `False` → `True` → `"$8"`

Example: Movie Ticket

```
1  age = int(input("Age: "))
2  is_student = input("Student? ")
3  if age < 12 or age >= 65 \
4     or is_student == "yes":
5      print("Discounted ticket: $8")
6  else:
7      print("Regular ticket: $15")
```

Trace 1: age=10, student="no"
True or False or False → True → "$8"

Trace 2: age=30, student="no"
False or False or False → False → "$15"

You Try

Write code to check if a number is between 1 and 100 (inclusive).

You Try

Write code to check if a number is between 1 and 100 (inclusive).

```
n = int(input("Enter a number: "))
if n >= 1 and n <= 100:
    print("Valid")
else:
    print("Out of range")
```

Common Mistakes

Pitfall #1: Missing Colon

✗ **Wrong:**

```
1 if x > 5
2       print("Big")
```

```
SyntaxError: expected ':'
```

Pitfall #1: Missing Colon

✗ **Wrong:**

```
1 if x > 5
2     print("Big")
```

```
SyntaxError: expected ':'
```

✓ **Fixed:**

```
1 if x > 5:
2     print("Big")
```

Always end `if`, `elif`, and `else` with a **colon** `:`

Pitfall #2: = vs ==

✗ **Wrong:**

```
1 x = 10
2 if x = 10:
3       print("Ten!")
```

Pitfall #2: = vs ==

✗ **Wrong:**

```
x = 10
if x = 10:
    print("Ten!")
```

`SyntaxError!` — = is **assignment**, not comparison!

Pitfall #2: = vs ==

✗ **Wrong:**

```
x = 10
if x = 10:
    print("Ten!")
```

`SyntaxError!` — = is **assignment**, not comparison!

✓ **Fixed:**

```
if x == 10:
    print("Ten!")
```

= assigns a value. == **compares** two values.

Pitfall #3: IndentationError

✗ Missing indent:

```
1 if x > 5:
2 print("Big")
```

IndentationError!

Pitfall #3: IndentationError

✗ Missing indent:

```
1 if x > 5:
2 print("Big")
```

IndentationError!

✗ Inconsistent indent:

```
1 if x > 5:
2     print("A")
3       print("B")
```

IndentationError!

Pitfall #3: IndentationError

✗ Missing indent:

```
1 if x > 5:
2 print("Big")
```

IndentationError!

✗ Inconsistent indent:

```
1 if x > 5:
2     print("A")
3       print("B")
```

IndentationError!

The body of an `if` **must** be indented. Use **4 spaces** consistently.

The `pass` Keyword

What if you want an empty body?

✗ **This crashes:**

```
1 if score >= 50:
2
3 else:
4     print("Failed")
```

The pass Keyword

What if you want an empty body?

✓ **Use** pass**:**

```
1 if score >= 50:
2     pass # TODO: implement later
3 else:
4     print("Failed")
```

pass = "do nothing" placeholder

Truthy / Falsy in Conditions

```
1 name = input("Enter name: ")
2 if name:
3     print(f"Hello, {name}!")
4 else:
5     print("You didn't enter a name!")
```

Truthy / Falsy in Conditions

```
1 name = input("Enter name: ")
2 if name:
3     print(f"Hello, {name}!")
4 else:
5     print("You didn't enter a name!")
```

Trace 1: `name = "Ali"` → truthy → `"Hello, Ali!"`
Trace 2: `name = ""` → falsy → `"You didn't enter a name!"`

Truthy / Falsy in Conditions

```
1 name = input("Enter name: ")
2 if name:
3     print(f"Hello, {name}!")
4 else:
5     print("You didn't enter a name!")
```

Trace 1: `name = "Ali"` → truthy → `"Hello, Ali!"`
Trace 2: `name = ""` → falsy → `"You didn't enter a name!"`

Falsy values: `0`, `0.0`, `""` (empty string), `None`
Everything else is truthy.

Chained Comparisons

Instead of:

```
1 if x >= 1 and x <= 10:
```

Chained Comparisons

Instead of:

```
1 if x >= 1 and x <= 10:
```

You can write:

```
1 if 1 <= x <= 10:
```

Chained Comparisons

Instead of:

```
1 if x >= 1 and x <= 10:
```

You can write:

```
1 if 1 <= x <= 10:
```

More examples:

- `if 0 < score < 100:` — score between 0 and 100
- `if 18 <= age <= 65:` — working age

Reads like math: $1 \leq x \leq 10$

You Try: Spot the Bugs!

Bug 1:

```
1 if x = 10:
2       print("Ten")
```

Bug 2:

```
1 if x > 5
2       print("Big")
```

Bug 3:

```
1 if x > 5:
2 print("Big")
```

Bug 4:

```
1 x = "5"
2 if x > 3:
3       print("Big")
```

You Try: Spot the Bugs!

Bug 1:

```
1 if x = 10:
2       print("Ten")
```

= should be ==

Bug 2:

```
1 if x > 5
2       print("Big")
```

Missing colon :

Bug 3:

```
1 if x > 5:
2 print("Big")
```

Bug 4:

```
1 x = "5"
2 if x > 3:
3       print("Big")
```

You Try: Spot the Bugs!

Bug 1:

```
1 if x = 10:
2       print("Ten")
```

= should be ==

Bug 2:

```
1 if x > 5
2       print("Big")
```

Missing colon :

Bug 3:

```
1 if x > 5:
2 print("Big")
```

Not indented

Bug 4:

```
1 x = "5"
2 if x > 3:
3       print("Big")
```

You Try: Spot the Bugs!

Bug 1:

```
1 if x = 10:
2       print("Ten")
```

= should be ==

Bug 2:

```
1 if x > 5
2       print("Big")
```

Missing colon :

Bug 3:

```
1 if x > 5:
2 print("Big")
```

Not indented

Bug 4:

```
1 x = "5"
2 if x > 3:
3       print("Big")
```

Comparing `str` to `int`!

TypeError at runtime

Practice

Trace Output #1

```
1  x = 7
2  y = 3
3  if x > 5:
4      print("A")
5      if y > 5:
6          print("B")
7      else:
8          print("C")
9  else:
10     print("D")
11 print("E")
```

Trace Output #1

```
1  x = 7
2  y = 3
3  if x > 5:
4      print("A")
5      if y > 5:
6          print("B")
7      else:
8          print("C")
9  else:
10     print("D")
11 print("E")
```

Trace:

7 > 5 → True → print "A"

3 > 5 → False → print "C"

then "E"

Trace Output #1

```
1  x = 7
2  y = 3
3  if x > 5:
4      print("A")
5      if y > 5:
6          print("B")
7      else:
8          print("C")
9  else:
10     print("D")
11 print("E")
```

Trace:

7 > 5 → True → print "A"

3 > 5 → False → print "C"

then "E"

Output: A, C, E

Trace Output #2

```
1  a = 10
2  b = 20
3  c = 30
4  if a > b:
5      print("X")
6  elif b > c:
7      print("Y")
8  elif a + b > c:
9      print("Z")
10 else:
11     print("W")
```

Trace Output #2

```
1  a = 10
2  b = 20
3  c = 30
4  if a > b:
5      print("X")
6  elif b > c:
7      print("Y")
8  elif a + b > c:
9      print("Z")
10 else:
11     print("W")
```

Trace:

10 > 20 → F, 20 > 30 → F

10+20 > 30 → 30 > 30 → F

Trace Output #2

```
1  a = 10
2  b = 20
3  c = 30
4  if a > b:
5      print("X")
6  elif b > c:
7      print("Y")
8  elif a + b > c:
9      print("Z")
10 else:
11     print("W")
```

Trace:

10 > 20 → F, 20 > 30 → F

10+20 > 30 → 30 > 30 → F

Output: `W`

Gotcha: 30 > 30 is `False` — it's **strict** greater than!

Write Code: Leap Year

A year is a leap year if:

- Divisible by 400 → Leap year
- Divisible by 100 → NOT leap
- Divisible by 4 → Leap year
- Otherwise → NOT leap

Write Code: Leap Year

A year is a leap year if:

- Divisible by 400 → Leap year
- Divisible by 100 → NOT leap
- Divisible by 4 → Leap year
- Otherwise → NOT leap

```
1 year = int(input("Enter year: "))
2 if year % 400 == 0:
3     print("Leap year!")
4 elif year % 100 == 0:
5     print("Not a leap year.")
6 elif year % 4 == 0:
7     print("Leap year!")
8 else:
9     print("Not a leap year.")
```

Write Code: Find Largest

Given three numbers,
print the largest.

Write Code: Find Largest

Given three numbers, print the largest.

```
1 a = int(input("a: "))
2 b = int(input("b: "))
3 c = int(input("c: "))
4 if a >= b and a >= c:
5     print(a, "is largest")
6 elif b >= a and b >= c:
7     print(b, "is largest")
8 else:
9     print(c, "is largest")
```

Fix the Code

✗ Buggy:

```
price = 250
if price >= 100:
    discount = "10%"
if price >= 200:
    discount = "20%"
if price >= 300:
    discount = "30%"
print(f"Discount: {discount}")
```

price=250 → matches 100 AND 200 → discount set to "10%" then "20%"

Fix the Code

✗ **Buggy:**

```
price = 250
if price >= 100:
    discount = "10%"
if price >= 200:
    discount = "20%"
if price >= 300:
    discount = "30%"
print(f"Discount: {discount}")
```

price=250 → matches 100 AND 200 → discount set to "10%" then "20%"

✓ **Fix:** Use `elif` + reverse order:

```
if price >= 300:
    discount = "30%"
elif price >= 200:
    discount = "20%"
elif price >= 100:
    discount = "10%"
else:
    discount = "0%"
```

Challenge: FizzBuzz

Given a number `n`:

- Div by 3 **and** 5 $\rightarrow$ `"FizzBuzz"`
- Div by 3 $\rightarrow$ `"Fizz"`
- Div by 5 $\rightarrow$ `"Buzz"`
- Otherwise $\rightarrow$ print `n`

Challenge: FizzBuzz

Given a number `n`:

- Div by 3 **and** 5 → `"FizzBuzz"`
- Div by 3 → `"Fizz"`
- Div by 5 → `"Buzz"`
- Otherwise → print `n`

```
1 n = int(input("Number: "))
2 if n % 3 == 0 and n % 5 == 0:
3     print("FizzBuzz")
4 elif n % 3 == 0:
5     print("Fizz")
6 elif n % 5 == 0:
7     print("Buzz")
8 else:
9     print(n)
```

Challenge: FizzBuzz

Given a number `n`:

- Div by 3 **and** 5 → `"FizzBuzz"`
- Div by 3 → `"Fizz"`
- Div by 5 → `"Buzz"`
- Otherwise → print `n`

```
1 n = int(input("Number: "))
2 if n % 3 == 0 and n % 5 == 0:
3     print("FizzBuzz")
4 elif n % 3 == 0:
5     print("Fizz")
6 elif n % 5 == 0:
7     print("Buzz")
8 else:
9     print(n)
```

Why must "both" come FIRST? If we check `% 3` first, 15 would print `"Fizz"` and stop!

Summary

Big Idea

Conditions let programs make
DECISIONS

Summary

- `if` — run code only when condition is True
- `if/else` — two paths, exactly one runs
- `elif` — multiple conditions, first match wins
- **Nested** `if` — an `if` inside another `if`
- `and`, `or`, `not` — combine conditions
- **Indentation** defines what's inside the body
- **Common pitfalls:** missing :, = vs ==, wrong indent

Coming Up Next...

Loops!

What if we want to repeat an action many times?

Coming Up Next...

Loops!

What if we want to repeat an action many times?

```
while condition:
```

```
for item in list:
```

Repeat code automatically!

Questions?

Exit Ticket

Today's question *(2 minutes)*

Let `x = 7` and run this chain:

```
if x > 10: print("A")
elif x > 5: print("B")
else: print("C")
```

(a) Which letter is printed?

(b) True or False: `x > 5 and x < 6`

COMP 102 — Exit Ticket #5

Name: _____

Roll #: _____ Date: _____

Answer:

Hand it to me on your way out.
No name, no attendance.